

CLAIMS

- 1 1. An apparatus comprising:
2 at least one processor;
3 a memory coupled to the at least one processor;
4 a first compilation unit residing in the memory, the first compilation unit
5 comprising a plurality of object oriented classes that are part of an object oriented
6 program, wherein the object oriented program is defined by the combination of the first
7 compilation unit and at least one other compilation unit; and
8 a compiler residing in the memory and executed by the at least one processor, the
9 compiler allocating at least one object in the first compilation unit to an invocation stack
10 frame for a method in the first compilation unit that allocates the at least one object.
- 1 2. The apparatus of claim 1 wherein the compiler comprises:
2 an escape analysis mechanism that marks each instruction in the first
3 compilation unit that allocates a new object as one of global escape, no escape,
4 and arg escape based on information available from classes visible in the first
5 compilation unit; and
6 an object allocation mechanism that allocates at least one object that is
7 created by an instruction marked as no escape by the escape analysis mechanism
8 to an invocation stack frame for a method that allocates the object.
- 1 3. The apparatus of claim 2 wherein the escape analysis mechanism marks each
2 instruction in the first compilation unit that allocates a new object as one of global escape,
3 no escape, and arg escape based on information available from classes visible in the first
4 compilation unit and from classes that are outside the first compilation unit that are
5 visible in a specified classpath.

- 1 4. The apparatus of claim 1 wherein the compiler comprises:
2 a code generator that creates two versions of code for a selected object method, a
3 first version using stack allocation of objects and a second version using heap allocation
4 of objects; and
5 a run time code selector that selects one of the first and second versions to execute
6 at run time based on a determination of whether classes seen at run time match expected
7 classes within predetermined limits.

1 5. An apparatus comprising:
2 at least one processor;
3 a memory coupled to the at least one processor;
4 a first compilation unit residing in the memory, the first compilation unit
5 comprising a plurality of object oriented classes that are part of an object oriented
6 program, wherein the object oriented program is defined by the combination of the first
7 compilation unit and at least one other compilation unit; and
8 a compiler residing in the memory and executed by the at least one processor, the
9 compiler comprising:
10 an escape analysis mechanism that marks each instruction in the first
11 compilation unit that allocates a new object as one of global escape, no escape,
12 and arg escape based on information available from classes visible in the first
13 compilation unit and from classes that are outside the first compilation unit that
14 are visible in a specified classpath;
15 an object allocation mechanism that allocates at least one object that is
16 created by an instruction marked as no escape by the escape analysis mechanism
17 to an invocation stack frame for a method that allocates the object;
18 a code generator that creates two versions of code for a selected object
19 method, a first version using stack allocation of objects and a second version
20 using heap allocation of objects; and
21 a run time code selector that selects one of the first and second versions to
22 execute at run time based on a determination of whether classes seen at run time
23 match expected classes within predetermined limits.

1 6. A method for allocating objects to memory in an object oriented program that
2 comprises a first compilation unit and at least one other compilation unit, the method
3 comprising the steps of:
4 (A) compiling the first compilation unit;
5 (B) during the compiling of the first compilation unit, allocating at least one
6 object that is created by an instruction in the first compilation unit to an invocation stack
7 frame for a method that allocates the at least one object.

1 7. The method of claim 6 wherein step (B) comprises the steps of:
2 marking each instruction that allocates a new object as one of global escape, no
3 escape, and arg escape based on information available from classes in the first
4 compilation unit and from classes that are outside the first compilation unit that are
5 visible in a specified classpath; and
6 allocating at least one object that is created by an instruction marked as no escape
7 by the escape analysis mechanism to an invocation stack frame for a method that allocates
8 the at least one object.

1 8. The method of claim 6 wherein step (B) comprises the steps of:
2 creating two versions of code for a selected object method, a first version using
3 stack allocation of objects and a second version using heap allocation of objects; and
4 selecting at run time one of the first and second versions to execute at run time
5 based on a determination of whether classes seen at run time match expected classes
6 within predetermined limits.

1 9. In an object oriented computer program that comprises a first compilation unit and at
2 least one other compilation unit, a method for allocating objects in the first compilation
3 unit to memory, the method comprising the steps of:
4 marking each instruction that allocates a new object as one of global escape, no
5 escape, and arg escape based on information available from classes in the first
6 compilation unit and from classes that are outside the first compilation unit that are
7 visible in a specified classpath;
8 creating two versions of code for a selected object method, a first version using
9 stack allocation of objects and a second version using heap allocation of objects; and
10 selecting at run time one of the first and second versions to execute at run time
11 based on a determination of whether classes seen at run time match expected classes
12 within predetermined limits.

1 10. A program product comprising:
2 a compiler that compiles a first compilation unit comprising a plurality of object
3 oriented classes that are part of an object oriented program, wherein the object oriented
4 program is defined by the combination of the first compilation unit and at least one other
5 compilation unit, the compiler allocating at least one object in the first compilation unit to
6 an invocation stack frame for a method in the first compilation unit that allocates the at
7 least one object; and
8 signal bearing media bearing the compiler.

1 11. The program product of claim 10 wherein the signal bearing media comprises
2 recordable media.

1 12. The program product of claim 10 wherein the signal bearing media comprises
2 transmission media.

1 13. The program product of claim 10 wherein the compiler comprises:
2 an escape analysis mechanism that marks each instruction in the first compilation
3 unit that allocates a new object as one of global escape, no escape, and arg escape based
4 on information available from classes visible in the first compilation unit; and
5 an object allocation mechanism that allocates at least one object that is created by
6 an instruction marked as no escape by the escape analysis mechanism to an invocation
7 stack frame for a method that allocates the object.

1 14. The program product of claim 13 wherein the escape analysis mechanism marks each
2 instruction in the first compilation unit that allocates a new object as one of global escape,
3 no escape, and arg escape based on information available from classes visible in the first
4 compilation unit and from classes that are outside the first compilation unit that are
5 visible in a specified classpath.

09865001.033404
T04287.0059860

1 15. The program product of claim 10 wherein the compiler comprises:
2 a code generator that creates two versions of code for a selected object method, a
3 first version using stack allocation of objects and a second version using heap allocation
4 of objects; and
5 a run time code selector that selects one of the first and second versions to execute
6 at run time based on a determination of whether classes seen at run time match expected
7 classes within predetermined limits.

- 1 16. A program product comprising:
 - 2 (A) a compiler that compiles a first compilation unit comprising a plurality of
 - 3 object oriented classes that are part of an object oriented program, wherein the object
 - 4 oriented program is defined by the combination of the first compilation unit and at least
 - 5 one other compilation unit, the compiler comprising:
 - 6 (A1) an escape analysis mechanism that marks each instruction that
 - 7 allocates a new object as one of global escape, no escape, and arg escape based on
 - 8 information available from classes in the first compilation unit and from classes
 - 9 that are outside the first compilation unit that are visible in a specified classpath;
 - 10 (A2) an object allocation mechanism that allocates at least one object that
 - 11 is created by an instruction marked as no escape by the escape analysis
 - 12 mechanism to an invocation stack frame for a method that allocates the object;
 - 13 (A3) a code generator that creates two versions of code for a selected
 - 14 object method, a first version using stack allocation of objects and a second
 - 15 version using heap allocation of objects; and
 - 16 (A4) a run time code selector that selects one of the first and second
 - 17 versions to execute at run time based on a determination of whether classes seen
 - 18 at run time match expected classes within predetermined limits; and
 - 19 (B) signal bearing media bearing the compiler.

- 1 17. The program product of claim 16 wherein said signal bearing media comprises
- 2 recordable media.

- 1 18. The program product of claim 16 wherein said signal bearing media comprises
- 2 transmission media.
